REMARKS

Claims 1-22 are currently pending in the subject application and are presently under consideration. Favorable reconsideration of the subject patent application is respectfully requested in view of the comments herein.

I. Rejection of Claims 1, 5-7, 13-17, and 19-22 Under 35 U.S.C. §103(a)

Claims 1, 5-7, 13-17, and 19-22 stand rejected under 35 U.S.C. §103(a) as being unpatentable over Bates *et al.* (U.S. Appln. No. 2003/0221185) in view of Bates *et al.* referred to as Bates_2 (U.S. 7,251,808). This rejection should be withdrawn for at least the following reasons. Neither Bates *et al.* nor Bates_2 teach or suggest all elements recited in the subject claims.

The subject application relates to debugging of computer processes and to facilitate attributed debugging. (*See e.g.,* pg. 1, lns. 9-10 and pg. 2, lns. 5-6.) The term "attributes" refers to keyword-like tags in which additional information about entities can be specified. (*See e.g.,* pg. 2, lns. 6-7 and pg. 6, lns. 6-9.) These attributes can be saved with an assembly's metadata and other applications, such as compilers, debuggers, can refer to the attribute information to determine how to use the objects (*e.g.,* evaluate the attribute…according to the attribute definition). (*See e.g., id.* at lns. 7-10 and pg. 6, lns. 9-11.)

Conventional systems provide challenges to developers due to the difficulty of displaying runtime object data in a simple and meaningful way. (*See e.g.,* pg. 2, lns. 11-16.) This display difficulty is due to the inability of the debugger to analyze arbitrary objects and determine what the developer would be most interested in viewing. (*See e.g., id.* at lns. 16-18.) The subject application allows a developer of an object to determine what will be displayed by the debugger, thus providing advantages over conventional systems. (*See e.g., id.* at lns. 19-20.)

Independent claim 1 (from which claims 5-7 and 13-16 depend) recites *a computer-implemented attributed debugging system, comprising ... an expression evaluator that* ***evaluates an attribute*** *associated with the computer software application* ***according to an attribute definition and presents debug information*** *associated with the computer software application* ***in accordance with the attribute definition,*** *wherein* ***the attribute definition declaratively indicates how the debug information is presented in a developer-customizable format.***

Claim 17 (from which claim 19 depends) recites similar features related to *a computer-implemented method facilitating attributed debugging, comprising ...**utilizing a definition of an attribute ... the attribute includes a declarative indication of how to display the debug information***.

Independent claim 20 recites *a data packet stored on computer readable media, the data packet transmitted between two or more computer components that facilitates debugging, the data packet comprising an attribute, **the attribute providing information associated with debugging of a computer software application, the information is presented in a developer-customizable format in accordance with the attribute information***.

Independent claim 21 recites similar features related to *a computer readable medium storing computer executable components of an attributed debugging system comprising ... **an expression evaluator ... that employs an attribute definition to evaluate an attribute... debug information is presented in a developer-customizable format and is presented in accordance with the attribute definition***.

Further, independent claim 22 recites similar features related to *a computer-implemented attributed debugging system comprising... means for employing a stored attribute and an attribute definition to present debug information ... wherein **the attribute definition declaratively indicates how the debug information is presented in a developer-customizable format***.

Attributes are employed to facilitate debugging of the application. (*See e.g.,* pg. 2, lns. 23-24.) An attribute definition can indicate how and/or whether a type or member is to be displayed in a data window. (*See e.g.,* pg. 9, ln. 9 to pg. 10, ln. 14). The attribute definition can be used to control what is presented for a given class or field in the data window. (*See e.g.,* pg. 10, ln. 16 to pg. 12, ln. 20.) Further, the attribute definition can be used to specify a display proxy of the type. (*See e.g.,* pg. 12, ln. 22 to pg. 13, ln. 11.) Since the attributes are an extensible, declarative way of conveying runtime information, the attributes provide a mechanism for allowing the developer to specify that behavior. (*See e.g.,* pg. 2, lns. 25-29).

In an example, the attributed debugging allows the manipulation of the view of data by allowing annotations that can be controlled. (*See e.g.,* pg. 2, lns. 30-31). This can include controlling what value should be shown at a top-level mitigating the need to expand the object for additional information. (*See e.g.,* pg. 3, lns. 1-2). It can also include controlling whether a

field or property should be presented, provide more descriptive information, and whether a type should be shown fully expanded. (*See e.g.,* pg. 3, lns. 5, 10, and 12-13.) This can also include controlling what value should be presented for a field or property, such as based on the result of an evaluation of an expression. (*See e.g.,* pg. 3, lns. 7-8). Neither of the cited references teach or suggest all features of the independent claims (and therefore the claims that depend there from).

Bates *et al.* relates to providing descriptions for variables while debugging or, more specifically to providing descriptive information (*e.g.,* in the form of text) for variables during debugging and to make comments and other information available to a user. (*See e.g.,* pg. 1, ¶[0003] and ¶[0009]; and pg. 2, ¶[0022] and ¶[0025].) Bates *et al.* discloses a debugger that determines whether any attributes are set for a variable. (*See e.g.,* pg. 6, ¶[0064].) These attributes can be set "off" or "on". (*See e.g.,* Fig. 3, elements 312 to 322 and pg. 4, ¶[0047].) The debugger determines whether any attributes are set for a variable, which are defined as machine-generated comments or use information. (*See e.g.,* pg. 2, ¶[0026] and pg. 6 [0064].) Examples of use information include G(global), S(static), I(index), P(parameter), R(return), and C(call), when may be displayed as fly-over text and which are represented as characters. (*See e.g.,* pg. 2, ¶[0026] and pg. 4, ¶[0047].) If these attributes are set the appropriate indicator is associated with the variable value. (*See e.g.,* pg. 6, ¶[0064]). Bates simply is nothing more than a filtration system that allows certain data to be included or eliminated from the view, but does not add anything to the debugging processes. Further, the comments and other information of Bates are not an attribute that is evaluated by an expression evaluator according to an attribute definition, as claimed.

Further, as conceded in the Office Action, Bates *et al.* is silent regarding wherein the attribute definition declaratively indicates how the debug information is presented in a developer-customizable format, as claimed, and Bates_2 is incorrectly relied upon to overcome the deficiencies of Bates *et al.*

Bates_2 relates to a graphical debugger with a loadmap display manager and a customer record display manager that displays user selected customized records from bound program objects. (*See e.g.,* Title.) The user can program the user interface to remember which fields of a variable type or variable are to be sent to an enhanced graphical user interface (GUI). (*See e.g.,* col. 3, lns. 49-52.) When a variable of that type is encountered, the GUI initially displays only the user-programmed fields for the variable or the record. (*See e.g.,* col. 3, lns. 52-55.)

However, this is not declaratively indicating how debug information is displayed in a developer-customized format, the debug information is presented in accordance with the attribute definition, as claimed.

Based on the above, the combination of Bates *et al.* and Bates_2 does not teach or suggest all elements recited in the independent claims (and thus the claims that depend there from). Accordingly, it is respectfully submitted that this rejection be withdrawn and the subject claims allowed.

II. Rejection of Claims 2-4 and 8-12 Under 35 U.S.C. §103(a)

Claims 2-4 and 8-12 stand rejected under 35 U.S.C. §103(a) as being unpatentable over Bates *et al.* in view of Bates_2 as applied to claims 1 and 7, and further in view of Dandoy (U.S. Appln. No. 2004/0230954). This rejection should be withdrawn for at least the following reasons. None of the cited references teach or suggest all elements recited in the subject claims.

The subject claims depend from independent claim 1. As discussed above, the combination of Bates *et al.* and Bates_2 do not teach of suggest *a computer-implemented attributed debugging system, comprising ... an expression evaluator that* ***evaluates an attribute ... according to an attribute definition and presents debug information ... in accordance with the attribute definition, ... the attribute definition declaratively indicates how the debug information is presented in a developer-customizable format.*** Dandoy relates to debugging a user interface but does not make up for the aforementioned deficiencies of Bates *et al.* and Bates_2.

Based on at least the above, the cited references do not teach or suggest all elements of independent claim 1, and therefore, cannot teach or suggest all elements of claims that depend there from. Thus, this rejection should be withdrawn and the subject claims allowed.

CONCLUSION

The present application is believed to be in condition for allowance in view of the above comments. A prompt action to such end is earnestly solicited.

In the event any fees are due in connection with this document, the Commissioner is authorized to charge those fees to Deposit Account No. 50-1063 [MSFTP578US].

Should the Examiner believe a telephone interview would be helpful to expedite favorable prosecution, the Examiner is invited to contact applicants' undersigned representative at the telephone number below.

Respectfully submitted,

AMIN, TUROCY & CALVIN, LLP

/Patrica S. Murphy/
Patrica S. Murphy
Reg. No. 55,964

AMIN, TUROCY & CALVIN, LLP
127 Public Square
57TH Floor, Key Tower
Cleveland, Ohio 44114
Telephone: (216) 696-8730
Facsimile: (216) 696-8731